Stata 7.0:
Funcionamento básico

Abrantes Araújo Silva Filho
Abril/2001

SUMÁRIO

1.1) INTRODUÇÃO AO STATA:.....3

1.2) INSTALAÇÃO E PROBLEMAS:.....3

1.3) INICIANDO O STATA E AJUSTANDO O DIRETÓRIO DE TRABALHO:.....3

1.4) AJUSTANDO A MEMÓRIA NA INICIALIZAÇÃO (E ON THE FLY):.....4

1.5) TERMINANDO O STATA:.....5

1.6) AMBIENTE DO STATA:.....5

1.7) USANDO O VIEWER:.....5

1.8) AJUDA:.....6

1.9) USANDO A INTERNET E ATUALIZANDO O STATA:.....6

1.10) RECURSOS PARA APRENDIZAGEM:.....7

1.11) -MORE-, "ERROR MESSAGES", "RETURN CODES", "RETURN MESSAGES", BREAK E USO DO TECLADO:.....8

1.12) LOGS: IMPRIMINDO E SALVANDO OUTPUTS.....8

1.13) AJUSTANDO FONTES E PREFERÊNCIAS DE JANELAS:.....9

1.14) SINTAXE DA LINGUAGEM:.....9

1.15) DADOS, FORMATOS, LABELS E NOTES:.....11

1.16) FUNÇÕES, EXPRESSÕES, VARIÁVEIS DO SISTEMA, ACESSO A COEFICIENTES E RESULTADOS, EXPLICIT SUBSCRIPTING, TIME-SERIES OPERATORS E PROBLEMAS DE PRECISÃO NO STATA:. . . .16

1.17) DO-FILES:.....21

1.18) ADO-FILES:.....21

1.19) COMANDOS IMEDIATOS E DISPLAY:.....22

1.1) Introdução ao STATA:

Referências: [U]3, [U]4, [R]about, [R]copyright

- O STATA é um software estatístico para manejo, análise e plotagem de dados.
- É ativado por comandos (command-driven).
- Filosofia é: digite um pouco, receba um pouco.
- É muito rápido devido a boa programação e armazenagem dos dados na memória: um dataset pode existir no disco, mas o STATA trabalha com uma cópia completa dos dados na memória. Isso aumenta a segurança: você pode adicionar, modificar, apagar ou fazer qualquer mudança nos dados na memória mas a única maneira de modificar/danificar permanentemente os dados é se você explicitamente salvar sobre o arquivo original. Manter todos os dados na memória também significa que o número máximo de observações é limitado pela quantidade de memória, assim, se você está com um grande banco de dados pode haver falta de memória (ver item 1.4 para saber como resolver o problema de memória).
- O STATA é disponível para diversos ambientes (windows, mac, unix, aix, solaris, linux,etc), mas a versão não importa: STATA é STATA e você pode compartilhar arquivos (datasets, graphs, programs, do-files, etc).
- STATA vem em dois "flavors": Intercooled e Small. Intercooled Stata é o STATA: é a versão profissional, sem limitações. Small Stata é uma versão para estudantes, limitada e mais lenta. Para determinar qual versão está instalada, digite o comando **about**. Para informações sobre o copyright, use o comando **copyright**.
- STATA é limitado a 2.047 variáveis, sendo que o número máximo de observações é limitado somente pela memória do computador

1.2) Instalação e problemas:

Referências: [GSW]1, [GSW]B, [U]6

- A licença do STATA dá direito a instalação em 2 computadores, para uso não simultâneo.
- O procedimento de instalação é simples e não modifica autoexec.bat ou config.sys.
- Para manter versões anteriores, especificar diretório diferente. Também será necessário especificar o diretório do STATA e o diretório padrão de trabalho (working directory).
- Quando for usar o STATA pela 1ª vez, será solicitado as informações da licença: nome, organização, serial number, code e authorization. É necessário completar todos os campos.
- Para verificar se a instalação foi correta, use o comando **verinst**. Lembre-se do comando **verinst**, pois se você mudar algo no setup do seu computador, você sempre poderá utilizar esse comando para saber se a instalação do STATA foi afetada.
- Se houver algum problema, verinst irá esclarecer o que está errado e como corrigir. Caso ocorram outros problemas (cannot find license file, error opening or reading the file, other messages, crashes) veja as FAQs ou contate o suporte.
- Após instalar, registre sua cópia do STATA em http://www.stata.com/register/

1.3) Iniciando o STATA e ajustando o diretório de trabalho:

Referências: [GSW]2, [U]5, [GSW]A, [R]cd

1.3.1) Iniciando o STATA através de atalhos e ajustando o diretório de trabalho:

- O STATA pode ser inicializado de diversas maneiras: menu iniciar, atalhos, duplo-clique em arquivos de dados, do-files, ado-files, smcl ou log files, etc; pode ser rodado no modo interativo (command-driven) ou em batch mode.
- O executável do STATA é o arquivo wstata.exe localizado geralmente no diretório c:\stata. Você pode criar vários atalhos para esse arquivo e controlar o modo como o STATA é inicializado em cada atalho: assim você pode criar várias modos de inicialização do STATA. O segredo é modificar as configurações do STATA na "janela propriedades":

  Tipo de objeto: Aplicativo
  Local do objeto: STATA
  Objeto: C:\STATA\WSTATA.EXE /m1
  Iniciar em: C:\STATA
  Tecla de atanho: Nenhum
  Executar: Maximizado

Comentário:
"Objeto" refere-se ao caminho do arquivo executável. A opção /m#, onde #=algum número, refere-se a quantidade de memória (em Mb) que é alocada ao STATA. "Iniciar em" refere-se ao diretório de trabalho (working directory). Esse diretório é onde serão salvos os dados, gráficos, logs, etc. O diretório de trabalho pode ser mudado através do comando **cd**:
cd ou **pwd** = mostra o diretório de trabalho
cd ["][*path*:*][directory_name]*["] = muda o diretório de trabalho

1.3.2) Executando comandos automaticamente ao inicializar o STATA:

- Você também pode determinar que certos comandos sejam executados toda vez que o STATA for inicializado. Para isso você deve criar um do-file com os comandos que você quer que o STATA execute na inicialização e salvá-lo com o nome ***profile.do*** no diretório de trabalho.
- O STATA não trabalha com mais de um dataset de uma vez

1.3.3) Executando em batch mode:

- Para executar um grande trabalho em batch mode, abra uma janela DOS, vá até o diretório de trabalho (working directory) e digite o comando:
  c:\stata\data> **c:\stata\wstata.exe [/m#] /{s|b} do *do-file***
  Esse comando diz ao STATA para executar o *do-file* em batch mode. A opção /b suprime todos os resultados e joga o output no arquivo *do-file.log* (formato ASCII); a opção /s faz a mesma coisa, mas joga o output no arquivo *do-file.smcl* (formato SCML).
  A opção /m# aloca a quantidade de memória, se necessário

1.3.4) Duplo-clique em dataset:

- Duplo-clique em um arquivo dataset (.dta) inicializa o STATA e abre o dataset. Para mudar a quantidade padrão de memória alocada quando inicializamos por duplo-clique em datasets, devemos modificar as opções na janela "tipo de arquivos". Para isso:
  a) Vá até meu computador; b) Clique no menu "ferramentas"; c) Clique no item "opções de pastas"; d) Clique na aba "tipos de arquivos"; e) Clique no tipo "DTA Stata Dataset"; f) Clique no botão "avançado"; g) Clique na ação "open"; h) Clique no botão "editar"; i) Modifique a opção /m# de acordo com o desejado; j) Clique em "OK", "OK" e "Fechar".
- Essa técnica também pode ser utilizada para que ao duplo-clique em um .dta, um ado-file seja executado e abra o arquivo (ver [GSW]A.10).

1.3.5) Duplo-clique em do-file ou arquivos gráficos:

- Duplo-clique em do-file (.do) inicializa STATA e executa o do-file. Para mudar a quantidade padrão de memória alocada quando inicializamos por duplo-clique em do-files, devemos mudar as opções na janela "tipo de arquivo". Para isso, ver procedimento para duplo-clique em datasets.
- Duplo-clique em arquivos gráficos (.gph) inicializa STATA e mostra o gráfico.

1.3.6) STATA multi-sessões:

- O STATA não trabalha com mais de 1 dataset por vez, mas é possível inicializar várias sessões simultâneas do STATA, cada uma trabalhando dom um dataset. Para isso é só inicializar outras sessões, por qualquer um dos meios discutidos anteriormente.

1.4) Ajustando a memória na inicialização (e on the fly):

Referências: [GSW]2, [GSW]A, [GSW]C, [R]compress, [R]memory, [R]query, [U]7

- O padrão após a instalação é que sejam alocados 1Mb para o uso do STATA. Se você trabalha com muitos dados, pode aumentar esse padrão alterando a opção /m# na "janela propriedades" do(s) atalho(s) que inicializa(m) o STATA.
- Atenção: o STATA é rápido pois trabalha com os dados na memória. Se você alocar muita memória, mesmo que não precise, o windows pode fornecer memória virtual, e isso diminui muito o desempenho do STATA. Quanto mais memória física (real), menor a chance do windows fornecer memória virtual (arquivo de troca).
- Para saber como está o uso da memória, utilize o comando **memory**. Verifique se é necessário aumentar a memória. Para aumentar a memória "on the fly", use o comando **set memory #{k|m}**, mas atenção: set memory só pode ser usado se não há dados na memória.
- Caso o banco de dados seja gigante, talvez seja necessário usar a memória

virtual. Nesse caso, para melhorar o desempenho do STATA quando ele usa a memória virtural, podemos utilizar o comando **set virtual {on|off}**. Atenção: o padrão é off, pois determinar "set virtual on" se na realidade não estivermos utilizando a memória virtual na realidade piora o desempenho do STATA. Para saber se set virtural está on ou off, utilize o comando **query**.

- Para tentar poupar ainda mais memória, pode-se usar o comando **compress *varlist***.

1.5) Terminando o STATA:

Referências: [GSW]A, [U]5, [R]exit

- Use o comando **exit [, clear]**.

1.6) Ambiente do STATA:

Referênias: [GSW]2, [U]5, [R]shell, [R]erase, [R]mkdir, [R]type, [R]cd, [R]copy, [R]dir

- STATA funciona em uma janela do windows, mas você pode chamar uma janela DOS ou inicializar um outro programa: use os comandos
  {shell|!} [*operating_system_command]*
  winexec *program_name* [*program_arguments*]
- STATA fornece 6 comandos DOS que evitam ter que ir ao shell ou digitar ! comando: erase, mkdir, type, cd, copy e dir. As sintaxes são:
  erase ["]*filename*["]
  dir ["]*filespec*["] [, wide]
  copy ["]*filename_1*["] ["]*filename_2*["] [, public text replace]
  cd ["][*drive:][directory_name]*["]
  type ["]*filename*["] [, asis showtabs]
  mkdir ["]*directory_name*["] [, public]
- Existe um add-on para o STATA 7.0 que modifica os menus, apresenta alguns utilitários e facilita a obtenção de algumas estatísticas para os iniciantes: é o STATAQUEST 7. Se estiver instalado, digite o comando **quest on|off** para ligá-lo/desligá-lo.

1.7) Usando o Viewer:

Referências: [GSW]3, [R]view

- A viewer é o local onde você pode:
  - ver e procurar informações de ajuda (geral, comandos, FAQ, keywords, STB, etc.)
  - ver e procurar informações de ajuda na internet (geral, comandos, FAQ, keywords, etc.)
  - ver (e imprimir) logs do seu trabalho atual ou passado
  - ver (e imprimir) qualquer arquivo SMCL ou ASCII
  - instalar e desinstalar novos comandos pela internet
  - atualizar o STATA pela internet
  - olhar um arquivo (.dta, .do, .ado, .log, .smcl, .txt, etc.) na internet
  - navegar por hyperlinks
  - iniciar seu browser padrão
  - ver as últimas notícias de http://www.stata.com/
- Para ir ao viwer:
  a) Clique no botão "Start viewer"; ou
  b) Digite o comando **view help viewer** na janela de comandos do STATA
- Para ver um arquivo ou um site da internet no viewer:
  a) Clique "File" -> "View..." e digite o nome do aquivo/URL ou clique em "browse" para procurar; ou
  b) Clique em "Start viewer" e clique no link "view your logs or any file"; ou
  c) Digite o comando **view [file] ["]*filename*["] [, asis]** na janela de comandos do STATA
- Para lançar seu browser padrão:
  a) Clique no link "launch your browser"; ou
  b) Na linha de comando do viewer, digite **browse *URL***; ou
  c) Na janela de comando do STATAm, digite **view browse ["]*URL*["]**

1.8) Ajuda:

Referências: [GSW]4, [U]8, [R]help, [R]view, [R]search

- STATA tem um excelente sistema de ajuda, compreendendo: Contents, Search, Stata

Command, e What's new

- Você pode visualizar a ajuda na janela de resultados (mais rápido) ou no viewer (melhor)
- Para ver o Contents:
  - Na janela de resultados: digite **help contents**
  - No viewer: digite **view help contents** ou
    clique "Help" -> "Contents" ou
    clique em "Start viewer" e no link "help for contents"
- Para procurar ajuda sobre assuntos via Keyword:
  - Na janela de resultados, use a seguinte sintaxe:
  search *word* [*word...*] [, author entry exact faq historical or manual stb]
  - No viewer: digite **view search *word* [*word...*]**
    clique em "Help" -> "Search" -> Marque "documentation and FAQ" -> busque
    clique em "Star viewer" e no link "search help files, doc..."
- Para ajuda sobre algum comando do STATA:
  - Na janela de resultados: **help *command***
  - No viewer: digite **view help *command*** ou digite **whelp *command***
    clique em "Help" -> "Stata command"
    clique em "Start viewer" e no link "help for any Stata command"
- Algumas considerações:
  - as seguintes palavras podem ser usadas para restringir o contexto:
    data
    statistics
    graph
    utility
    programs
  - capitalização, pontuação, ordem das palavras e preposições (for, to, etc.) não importam
  - é permitir abreviar (melhor que digitar errado)
  - evitar plurais
  - quando visualizamos a ajuda na janela de resultados e o **-more-** surge, podemos teclar a barra de espaços para ir para a próxima tela, e a letra b para voltar a tela anterior

1.9) Usando a internet e atualizando o STATA:

Referências: [GSW]20, [U]32, [R]news, [R]net, [R]net search, [R]update, [R]stb

- Para saber as últimas notícias sobre o STATA, bem como as atualizações lançadas:
  - Na janela de resultados: **news**
  - No viewer: clique em "Help" -> "News"; ou
    digite o comando **view news**; ou
    clique em "Start viewer" e no link "see the lateste news" ou no botão "news"
- As atualizações/adições ao STATA são de 3 formas:
  a) Atualizações oficiais: são fornecidas pela STATA Coorporation e compreendem atualizações do executável do STATA e dos ado-files (comandos do STATA. Ver 1.18)
  b) User-written additions: são ado-file (comandos do STATA) escritos por diversos usuários no mundo inteiro. Geralmente são distribuídas através do STATA Technical Bulletin (STB) mas podem ser de outras formas (outros web-sites, disquetes, etc.)
- Para baixar as atualizações oficiais (STATA executável e ado-files da STATA Corp.):
  - Na janela de resultados: **update**
    update from *location*
    update query [, from(*location*)]
    update ado [, from(*location*) into(*dirname*)]
    update executable [, from(*location*) into(*dirname*) force]
    update all [, from(*location*)]
  - No viewer: clique em "Help" -> "Official updates"
    digite **view update** e siga as instruções (comandos "update" são os mesmos)
    clique em "Start viewer" e no link "Official updates"
- Para procurar e instalar os STB e programas escritos por usuários:

- Na janela de resultados:
  Para procurar:
    net search *keywords* [, or nostb tocpkg toc pkg everywhere filename]
  Para navegar:
    net
    net from *directory_or_URL*
    net cd *path_or_URL*
    net cd ..
    net link *linkname*
  Para descrever:
    net describe *pkgname*
  Para configurar diretórios de gravação:
    net query
      net set ado *dirname*
    net set other *dirname*
  Para instalar os .ado e .hlp:
    net install *pkgname* [, all replace]
  Para baixar as "ancillary files":
    net get pkgname [, all replace]
  Para achar, descrever e desinstalar os .ado instalados:
    ado [, find(*string*) from(*dirname*)]
    ado dir [pkgid] [, find(*string*) from(*dirname*)]
    ado describe [pkgid] [, find(*string*) from(*dirname*)]
    ado uninstall pkgid [, from(*dirname*)]
- No viewer: clique em "Help" -> "Search" -> marcar "net resources" -> pesquisar
  clique em "Start viewer" e no link "find and install STB and user..."
  digite **view net** e procure nos diretórios

1.10) Recursos para aprendizagem:

Referências: [GSW]19, [U]2, [U]9

- STATA tem o melhor suporte ao usuário e recursos de aprendizagem entre todos os programas de estatística. Entre eles:
  a) STATA web site: http://www.stata.com = fornece diversos materiais, FAQs, adições e links para muitos sites de aprendizagem;
  b) STATA listserver: lista de discussão na internet, extremamente útil
  c) STB: STATA Technical Bulletin
  d) Atualizações e upgrades pela internet
  e) NetCourses
  f) Livros e outros materiais de suporte (ver STATA web site)
  g) Suporte técnico via fax, telefone e e-mail
- STATA vem com 11 tutoriais on-line (preferivelmente devem ser executados após a leitura completa do Getting Started for Windows). Para ativar, digite **tutorial *filename*,** onde os nomes dos arquivos dos tutoriais são:
  intro.tut
  contents.tut
  graphics.tut
  tables.tut
  regress.tut
  anova.tut
  logit.tut
  survival.tut
  factor.tut
  ourdata.tut
  yourdata.tut
- STATA também vem com uma série de datasets de exemplo.

1.11) -more-, "error messages", "return codes", "return messages", break e uso do teclado:

Referências: [GSW]10, [U]11, [U]10, [U]12, [U]13, [R]#review, [R]more

- Quando surge o **-more-** na janela de resultados, podemos teclar:

- enter = próxima linha
- espaço = próxima tela
- q = break
- botão "Clear -more- condition"
- botão "Break"
- no próprio **-more-** = próxima tela

- Para ligar/desligar o **-more-** usar o comando: **set more on|off**
- Quando ocorre um erro, STATA retorna uma mensagem de erro e um código de retorno (error messages and return codes). Os return codes são clicáveis: mostram o que significam.
- Sempre que ocorrer um erro em um .ado ou .do, STATA imediatamente para a exeucução dos comandos e mostra a "error message" e o "return code" apropriados.
- STATA também pode fornecer uma "<u>return message</u>", que indica se o comando foi completado com sucesso, o tempo, em segundos, que o STATA gastou para executar o comando, e a hora em que o comando foi executado. Para ligar/desligar as "return message" use o comando: **set rmsg on|off**
- Quando clicamos em "Break" (ou usamos ctrl+break), STATA pára o que estava fazendo, desfaz o que já tinha feito e espera um comando. O estado das coisas é o mesmo que nunca ter dado o comando que break parou, com exceção de: a) se você estava lendo dados com insheet, infile ou infix, o que o STATA já tinha lido é mandito na memória; e b) se você estava utilizando o sort em uma segunda (ou +) variável, após o break todos os sort anteriores são zerados.
- STATA fornece a possibilidade de usar e configurar as teclas F1-F12 para acionar comandos (com exceção da F10 que é utilizada pelo windows). As teclas F que são padrão(; = enter):
  F1 = help
  F2 = #review;
  F3 = describe;
  F7 = save
  F8 = use
- Para configurar qualquer tecla F, use o comando: **global F# "*command*[;]"**
- Para zerar a configuração de qualquer tecla F, use: **global F#**
- Para ver linhas anteriores, pode-se usar a janela review ou usar o comando **#review [#1 [#2]]** (note que o comando #review sempre mostra a partir do mais recente)
- Para evitar tempo de digitação, os nomes das variáveis e dos comandos podem ser abreviados. A menor abreviação permitida em cada comando é mostrada por um sublinhado em sua sintaxe. A menor abreviação permitida para cada variável é aquela que permite uma identificação única de cada variável

1.12) Logs: imprimindo e salvando outputs

Referências: [GSW]17, [U]18, [R]log, [R]translate

- Um log é o registro de todo seu trabalho no STATA, e pode ser armazenado de 2 modos: SMCL (Stata Markup and Control Language) ou LOG. Um log SMCL é uma cópia perfeita de tudo o que saiu na janela resultados. Um log LOG é uma cópia em formato ASCII puro do que saiu na janela resultado. Somente o STATA entende SMCL: assim você somente poderá ver e imprimir logs SMCL através do STATA. Logs LOG (ASCII) são compreendidos por diversos programas. Você pode transformar um log SMCL em LOG (ASCII).
- Para iniciar um log:
  - clique no botão "Begin, suspend, close logs", escolha SMCL ou LOG e dê um nome
  - clique "File" -> "Log" -> "Begin"
    log
    [quietly] log using *filename* [, append replace [<u>t</u>ext|<u>s</u>mcl]]
    set logtype {<u>t</u>ext|<u>s</u>mcl}
- Para ver o log:
  - clique "File" -> "Log" -> "View"
  - clique "Begin, suspend, close logs" e escolha "view snapshot..."
- Para temporariamente suspender ou recomeçar o log:
  - clique no botão "Begin, suspend, close logs", escolha "suspend" ou "resume"
  - clique "File" -> "Log" -> "suspend" ou "resume"
    log {off|on}
- Para terminar o log:

- clique no botão "Begin, suspend, close logs" e escolha "close log"
- clique em "File" -> "Log" -> "Close"
  [quietly] log close

- Para traduzir um log SMCL em LOG (ASCII)
  - clique em "File" -> "Log" -> "Translate"
  translate *filename*.smcl *filename*.log
- Para fazer um log somente com os comandos, use:
  cmdlog
  cmdlog using *filename* [, append replace]
  cmdlog {on|off|close}
  ou clique no botão esquerda da janela review e clique "Save review contents"
- O uso do log e do cmdlog pode ser simultâneo. Para incluir comentários, inicie a linha com um *
- O uso dos logs, comentários * em logs, e do notes (ver a seguir) é uma maneira muito eficiente de documentar toda a análise, desde a entrada de dados, até os resultados finais. Atenção: **USE SEMPRE** os recuros de logs, comentário e notes.

1.13) Ajustando fontes e preferências de janelas:

Referências: [GSW]18

- As fontes podem ser mudadas nas seguintes janelas: resultados, gráficos, viewer, comandos, review, variables, data editor, do-file editor. Dê preferência para fontes de tamanho fixo. Para mudar a fonte para uma determinada janela, clique no ícone da barra de título e selecione "Fonts". Para ajustar as fontes do do-file editor, clique em "Edit" -> "Preferences".
- Para salvar as modificações no "windowing", clique "Prefs" -> "Save windowing preferences".

1.14) Sintaxe da linguagem:

Referências: [U]14, [R]by, [U]31.2

1.14.1) Introdução:

- A sintaxe geral dos comandos do STATA é:
  [by *varlist1* [(varlist2)] [, sort rc0]:] command [*varlist*] [=*exp*] [if *exp*] [in *range*] [*weight*] [, *options*]
- [*varlist*] = indica as variáveis que sofrerão o efeito do comando. Se não for especificado, os comandos assumem varlist = **_all** (comandos destrutivos necessitam obrigatoriamente da especificação de variáveis)
- [by *varlist*:] = indica que o comando deve ser repetido para cada grupo de valores da *varlist*. Atenção: para poder usar [by varlist:] os dados devem estar organizados com o comando **sort**. Quando usamos [by *varlist1 (varlist2)* [, sort]], queremos dizer que o STATA primeiro certifique que varlist1 e varlist2 estão organizados e, se a condição for "true", executa o by pela varlist1. by varlist trabalha alocando, em cada interação, os valores das variáveis de sistema _n (observação atual) e _N (total de observações) em relação à 1ª observação (_n) de cada by-group.
- [if *exp*] = restringe o escopo do comando às observações para as quais o valor da *exp* é "true" (equivalente a expressão ser não zero).
- [in *range*] = restringe o escopo do comando a um determinado "range" de observações. Sua sintaxe correta é: [**in #1/#2**], lendo-se "da observação #1 à observação #2", sendo que podem ser números negativos (indicam contagem a partir do final), f = first e l = last. Atenção: [in *range*] não pode ser usado com [by *varlist*[, sort]], mas essa limitação pode ser superada usando-se as variáveis _n em if=exp, por exemplo: if _n<=3
- [=*exp*] = especifica valores a serem definidos a uma variável.

- [*weight*] = indica o peso a ser atachado a cada observação. Sua sintaxe correta é:
  [*weightword=exp*], onde "weightword" pode ser um dos quatro tipos de pesos que o STATA entende: **fweight** ou **frequency** (indicam observações duplicadas), **pweight** ("sampling weight": denotam o inverso da probabilidade de que essa observação seja incluída na amostra devido ao desenho da amostragem), **aweight** ou **cellsize** ("analytic weights": são inversamente proporcionais à variância de uma observação), **iweight** ("importance weights": indicam a importância relativa de

cada observação). **weight** pode ser usado como wheightword, mas significa que o comando é que deve determinar qual dos 4 tipos de pesos será usado (o output informa qual o tipo que o STATA determinou que seria úitl). Quando não especificamos um "weight", o resultado é equivalente a especificar "fweight=1".

- [, *options*] = são opções específicas de cada comando. Somente uma vírgula deve existir: a que separa a sintaxe básica das opções (não é permitido vírgulas entre as opções). Algumas opções necessitam de argumentos, que devem ser escolhidos por você. Quando uma oção necessita de um argumento, ele é marcado entre parênteses, por exemplo: , n(#). Algumas opções podem necessitar de mais de 1 argumento: nesse caso os argumentos devem ser separados uns dos outros por vírgulas, por exemplo: save(*filename*[, replace]). Obs.: listas, tais como listas de variáveis (varlists) e listas de números (numlists) são consideradas como um único argumento, e os componentes das listas não devem ser separados por vírgulas. Por exemplo: se a sintaxe é: ylabel(*numlist*), numlist é considerado um argumento único e portanto seus componentes não devem ser separados por vírgulas, ficando ylabel(1 2 3 4). Algumas opções usam strings: tudo o que foi dito sobre as opções e argumentos vale para strings, com a única diferença de que a string deve vir entre aspas, por exemplo: title("*text*").

1.14.2) Varlists:

- Uma *varlist* é uma lista de variáveis, e deve se referir exclusivamente a variáveis novas (ainda não criadas) ou exclusivamente a variáveis já existentes.
- Quando listamos variáveis já existentes, elas podem ser repetidas. Também podemos usar símbolos especiais para aberviar os nomes das variáveis:

  a) ***** para indicar "zero ou mais caracteres a partir daqui", sendo que pode ser usado no início, meio ou fim do nome de uma variável, e pode ser colocado mais de uma vez em cada nome de variável,por exemplo:

  apgar* = lista todas as variáveis que começam com apgar
  i*95 = lista todas as variáveis que começam com i e terminam com 95
  *te = lista todas as variáveis que terminam com te
  in*cd*98 = lista todas as variáveis que começam con in, tem cd no meio e termina com 98
  *cd*re = lista todas as variáveis com cd no meio e terminam com re

  b) **~** também serve para indicar "zero ou mais caracteres a partir daqui", mas difere de * no sentido de que ~ especifica que existe apenas 1 variável que cumpre o padrão especificado. O resto é semelhante a *. Se o uso de ~ não especifica unicamente uma variável, STATA retorna uma mensagem de erro "ambiguous abbreviation".

  pe~ = lista a variável que começa com pe; se há mais de uma variável que começa com pe, STATA irá retornar uma mensagem de erro dizendo: "ambiguous abbreviation"
  i~93~casa = lista a variável que começa com i, tem 93 no meio e termina com casa; se há mais de uma variável assim, STATA irá retornar mensagem de erro.
  ~zhe = idem;
  dfer~df = idem

  c) **?** serve para substituir apenas 1 caractere:

  apgar?
  casa?ento
  c?af??mer?

  d) **-** serve para listar todas as variáveis entre as duas listadas, inclusie. A listagem é mostrada na ordem que estão no dataset, e não em ordem alfabética. Se seu dataset contém as variáveis v1 v4 v19 v2 v3 v10 e você especificar v1-v19, serão listadas apenas as variáveis v1, v4 e v19.

  e) Observação: todos os símbolos *****, **~**, **?** e **-** podem ser usados simultaneamente e de todas as maneiras possíveis.
- Quando listamos variáveis novas (ainda não criadas), elas não podem ser repetidas e não podemos usar símbolos de abreviação. O único símbolo que pode ser usado é um - entre variáveis que começam com a mesma letra e terminam em números. Assim, por exemplo, v1-v4 significa v1 v2 v3 v4.
- STATA permite "time-series varlists". Ver [U]14.4.3

1.14.3) Convenções de arquivos:

- Os nomes de arquivos podem ser dados de acordo com o padrão do windows, mas as extensões são as seguintes:

.ado = ado-files
.dct = ASCII data dictionary
.do = do-files
.dta = dataset
.gph = graph image
.log = ASCII log
.smcl = SMCL log
.out = arquivo salvo pelo comando **outsheet**
.raw = ASCII-formated dataset
.sum = arquivo checksum para verificar transeferências via rede

1.14.4) Convenções de nomes:

- Um nome é uma seqüência de 1-32 letras (A-Z e a-z), dígitos (0-9) e subscritos (_). O STATA reserva os seguintes nomes, pois são variáveis de sistemas ou outros:

_all	double	long	_rc
_b	float	_n	_se
byte	if	_N	_skip
_coef	in	_pi	using
_cons	int	_pred	with

- STATA respeita capitalização (maiúsculas/minúsculas). Não é recomendado iniciar nomes de variáveis com subscrito pois as variáveis internas do STATA (built-in variables) começam com _, e é melhor usar minúsculas e não acentuar, para facilitar a digitação.
- Para salvar nomes de arquivos que contenham espaços ou para evitar que o windows coloque uma segunda extensão no nome do arquivo, use aspas.
- O nome de uma variável tem de 1 a 32 caracteres, e não pode conter espaços.

1.15) Dados, formatos, labels e notes:

Referências: [GSW]18, [U]15, [R]format, [U]16.10, [GSW]8, [R]label, [R]notes

1.15.1) Dados e datasets:

- Um dataset do STATA compreende os dados, formatos (dos dados), labels, notes e charecteristics.
- Os dados podem ser números, strings (texto), elapsed dates ou time-series. (na realidade somente existem dois tipos de dados, números e strings: os formatos elapsed date e time-series são, na verdade, tipos numéricos com formatação especial).

1.15.2) Números

- Uma variável numérica pode conter até um sinal, uma parte inteira, uma parte decimal, a letra e ou E, o sinal do expoente e o expoente. Não podem conter vírgulas: o separador decimal é um ponto (apesar do STATA poder mostrar vírgulas como separador). Por exemplo: -5.32e-4, 1,000,000.00, 203948,345
- Existem 5 tipos de variáveis numéricos:

tipo	mínimo	máximo	+perto de 0	missing	significado
byte	-127	126	+-1	127	inteiro
int	-32767	32766	+-1	32767	inteiro
long	-2147483647	2147483646	+-1	2147483647	inteiro
float	-10e36	10e36	+-10e-36	2e128	real 8.5
double	-10e308	10e308	+-10e-323	2e1023	real 16.5

- Um número pode ser missing. No STATA é representado por um ponto: .
  Note também que um valor numérico missing não é salvo como um "vazio" e sim como o maior valor permitido pelo tipo de dado numérico em particular (ver tabela). Isso leva a 4 fatos importantes:
  a) Se você organizar os valores em ordem ascendente, os valores missing ficarão em último lugar (pois são gravados como os maiores valores que o tipo de dado permite);
  b) Se você usar if=*exp*, deve ter sempre em mente os valores missing, pois, por exemplo, o comando "list if income > 400" também irá listar os missing.
  c) Qualquer operação aritimética, lógica ou outra com um valor missing, retorna um missing (e o STATA informa quantos missing foram gerados)
  d) No output e nos cálculos, o STATA ignora os valores missing. Alguns comandos irão descartar toda a observação (casewise deletion) se uma ou mais das variáveis da obserção for missing.
- Atenção: existe uma "falha" no STATA quando utilizamos variáveis tipo float: todos os cálculos e expressões numéricas em STATA são realizados com dados tipo

double. Mesmo se o seu dado for float ele será "transformado" temporariamente para double para os cálculos. Isso gera um problema pois, por exemplo, o número 1.1 em formato float é diferente do número 1.1 em formato double (uma diferença de mais ou menos 10e8). Assim, por exemplo, se você tem uma var1 em formato float com o valor de 1.3, e der um comando tipo: count if var1==1.3 o resultado será erroneamente 0: isso ocorre pois a expressão é avaliada em formato double e o dado original está em formato float (10e8 menor), sendo então diferentes. Para evitar esse problema é melhor que todas as suas variáveis numéricas, não inteiras, sejam do formato double. Isso tem a desvantagem de consumir mais memória. Caso você queira manter seus dados em formato float, sempre que precisar usar um expressão de igualdade (ou outras em que haja esse problema), poderá usar a função float() (ver item 1.16.2). Para as variáveis que vão armazenar apenas números inteiros (positivos ou negativos), podem ser utilizados os formatos byte, int ou long sem problemas (isso é melhor pois poupa muita memória): lembre-se que o problema citado nesse parágrafo só ocorre quando armazenamos dados não inteiros (tipo 3.4, -545,234) em formato float, por isso é melhor armazená-los como double. Para maiores informações ver: [U]16.10.

1.15.3) Strings:

- As variáveis string no STATA respeitam a capitalização, ou seja, respeitam maiúsculas e minúsculas.
- Durante comandos e análises, quando nos referimos a uma determinada string, devemos colocá-la entre aspas. Durante a entrada de dados no data editor, ao contrário, não é necessário colocar entre aspas.
- Strings também permitem valores missing, armazenados como "". Um ponto não é interpretado como missing, mas como um ponto mesmo: "."
- Variáveis string não podem ser usadas diretamente nos cálculos estatísticos. Por exemplo, se você tem uma variável string 1=homem e 2=mulher, muitos comandos estatísticos não entenderão como tratar essa variável string e fornecerão uma mensagem de erro ou dirão que todas as observações são missing. Para os cálculos, é necessário transformas as string 0 e 1 em números 0 e 1 (os comandos encode e decode servem para isso).
- Se uma variável string contém somente a representação de números, p. ex.: "1.2", "5", etc., ela pode ser convertida diretamente em uma variável numérica pela função real() (o inverso é feito pela função string()).
- Existem 80 tipos de variáveis string, definidas de acordo com o número de caracteres que suportam: de 1 a 80:

tipo	significado
str1	string com 1 caractere
str2	string com 2 caracteres
str3	string com 3 caracteres
...	...
str80	string com 80 caracteres

1.15.4) Datas:

- Você pode registrar datas da maneira que quiser, mas STATA somente entende uma forma: "elapsed dates". Uma elapsed date é o número de dias a partir de 01/jan/1960. Assim:

elapsed data	significado
0	1/jan/1960
1	2/jan/1960
31	1/fev/1960
12784	1/jan/1995
-12784	31/dez/1924
-2	30/dez/1959

- STATA pode registrar datas de 01/jan/100 até 31/dez/9999
- Como você registra datas da maneira que quiser e STATA somente entende elapsed dates, STATA fornece funções para converter datas em elapsed dates.

1.15.5) Time-series:

- Além das elapsed dates, STATA fornece 5 outros formatos para datas: são as chamadas variáveis time-series (time-series variables) (funcionam de modo semelhando às elapsed dates):

weekly:	-1	=	1959, semana 52
	0	=	1960, semana 1

```
                1    =     1960, semana 2
monthly:       -1    =     1959, dezembro
                0    =     1960, janeiro
                1    =     1960, fevereiro
quarterly:     -1    =     1959, 4° trimestre
                0    =     1960, 1º trimestre
                1    =     1960, 2º trimestre
half-yearly:-1       =     1959, 2º semestre
                0    =     1960, 1° semestre
                1    =     a960, 2° semestre
yearly:        -1    =     1959
                0    =     1960
                1    =     1961
```

1.15.6) Formatos controlam como os dados são mostrados:

- O modo como os dados são mostrados no output, nos comandos, etc., é controlado independentemente de como os dados são armazenados (o exemplo mais claro é com as datas: STATA somente entende elapsed dates e uma data registrada como 0, pode ser mostrada no output como 01/jan/2001 ou outro formato). STATA fornece total controle sobre como formatar os dados para serem mostrados.
- **ATENÇÃO**: Para as variáveis numéricas e de string, a amplitude dos formatos (ver a seguir) deve ser no mínimo igual ao dos nomes das variáveis ou do maior valor registrado (o que for maior), para que o STATA possa fazer listagens e tabelas corretamente. Se essa regra não for seguida, as listagens e tabulações apresentadas pelo STATA poderão estar "desalinhadas", prejudicando a legibilidade dos resultados.

1.15.6.1) Formatos numéricos:

- O formato de exibição de uma variável numérica é dado por:
  format *varlist* %[-]w.d{e|f|g}[c]
  % = obrigatório para indicar início do formato
  [-] = opcional para indicar alinhamento à esquerda
  w = amplitude do formato (deve prever sinais, notação exponencial, etc.)
  . = obrigatório para indicar a casa decimal como . ou como ,
  d = número mínimo de casas decimais (deve ser menor do que w)
  {e|f|g} = são os 3 formatos numéricos do STATA (apenas 1 deve ser escolhido):
  e = notação científica
  f = fixo
  g = geral (o STATA escolhe o formato baseado nos valores da variável)
  [c] = vírgulas opcionais para separação de milhares (não permitido se formato for e)
- Por padrão, toda variável numérica tem o formato %w.0g, onde w é grande o suficiente para mostrar o maior número do tipo da variável.
- O formato g é um complicado conjunto de regras que tenta automaticamente mostrar os valores de um modo legível sem sacrificar a precisão numérica.
- O formato padrão das variáveis numéricas é:

byte	%8.0g
int	%8.0g
long	%12.0g
float	%9.0g
double	%10.0g

- Atenção: por padrão, o STATA não entende dados numéricos registrados com vírgulas para separar as casas decimais: apenas . como separador decimal é permitido na entrada de dados. Mas para mostrar os dados, podemos formatar os valores numéricos para mostrar . como separador de milhares e , como separador decimais. Isso é feito com o comando:
  set dp {comma|period} (embora isso não seja recomendado pois alguns comandos antigos do STATA podem não funcionar corretamente se set dp comma.
- Lembre-se que a amplitude do formato deve ser no mínimo igual ao dos nomes das variáveis ou do maior valor registrado (o que for maior), para que o STATA possa fazer listagens e tabelas corretamente. Se essa regra não for seguida, as listagens e tabulações apresentadas pelo STATA poderão estar "desalinhadas", prejudicando a legibilidade dos resultados.

1.15.6.2) Formatos de strings:

- O formato de uma variável string é dado por:
  format *varlist* %[-]ws
  % = obrigatório para indicar início do formato
  [-] = - opcional para indicar alinhamento à esquerda
  w = amplitude do formato
  s = obrigatório para indicar fim do formato
- Lembre-se que a amplitude do formato deve ser no mínimo igual ao dos nomes das variáveis ou do maior valor registrado (o que for maior), para que o STATA possa fazer listagens e tabelas corretamente. Se essa regra não for seguida, as listagens e tabulações apresentadas pelo STATA poderão estar "desalinhadas", prejudicando a legibilidade dos resultados.

1.15.6.3) Formatos de datas:

- Formatos de datas são, na realidade, formatos numéricos pois o STATA armazena datas como elapsed dates. A sintaxe é <u>muito complexa</u>:
  format *varlist* %[-]d[*]
  % = obrigatório para indicar início do formato
  [-] = - opcional para indicar alinhamento à esquerda
  d = obrigatório para indicar formato de data
  * = indica de 0 a 11 outros caracteres para formatar corretamente as datas
- Os caracteres * são (note que algumas opções obrigatoriamente excluem outras):
  d/D mostra o dia do mês 1-31 sem/com 0 anterior
  m/M mostra o mês, 1ª letra maiúscula, em abreviatura de 3 letras (m) ou extenso (M)
  l/L mostra o mês, 1° letra minúscula, em abreviatura de 3 letras (l) ou extenso (L)
  c/C mostra o século sem/com um 0 anterior
  y/Y mostra o ano com dois dígitos sem/com 0 anterior
  n/N mostra o número do mês 1-12 sem/com 0 anterior
  j/J mostra o dia do ano 1-366 sem/com 0 anterior
  h mostra o número do semestre, 1 ou 2
  q mostra o número do trimestre, 1, 2, 3 ou 4
  w/W mostra o número da semana no ano 1-52 sem/com 0 anterior
  _ mostra um espaço (_ = subscrito)
  . mostra um ponto
  , mostra uma vírgula
  : mostra dois pontos
  - mostra um traço
  / mostra uma barra
  ' mostra uma aspa única
  !c mostra um ponto de exclamação
- Especificar somente %d é equivalente a %dDlCY (ex.: 01jan1999)
- Note que o formato dd/Mmm/yyyy é obtido por %dD/m/CY. Note também que com combinações dos caracteres *, diversos formatos para mostrar datas podem ser conseguidos.
- Note também que é melhor escrever as datas no formato dd, mmm e yyyy, ou seja, formatando o mês para a abreviatura de 3 letras ao invés de números. Isso é melhor pois evita a confusão de datas no formato americano e europeu.

1.15.6.4) Formato das time-series:

- Da mesma forma que os formatos das datas, os formatos das time-series são formatos numéricos. A sintaxe é:
  format *varlist* %[-]t{d|w|m|q|h|y}[*]
  % = obrigatório para indicar início do formato
  [-] = - opcional para indicar alinhamento à esquerda
  t = obrigatório para indicar formato de time-series
  {d|w|m|q|h|y} = especifica como a data é codificada
  d se 0 = 01jan1960 (mesmo que %d)
  w se 0 = 1960w1 (ano 1960, semana 1)
  m se 0 = 1960m1 (ano 1960, mês 1)
  q se 0 = 1960q1 (ano 1960, trimestre 1)
  h se 0 = 1960h1 (ano 1960, semestre 1)
  y se 0 = 1960 (ano 1960)
  * = indica de 0 a 11 outros caracteres para formatar corretamente as datas
- Os caracteres * são os mesmos para os formatos de datas (ver item 1.15.6.5)

1.15.7) Labels:

- Labels são strings usados para "nomear" coisas. STATA fornece labels para datasets, variáveis e valores. Infelizmente STATA não consegue entender caracteres acentuados em português, portanto é melhor não usar acentuação de nenhum tipo.

- Dataset labels: associado a cada dataset existe um label (inicialmente em branco) de 80 caracteres. Para utiliá-lo utilize o comando:
  label data "*label*"
  Toda vez que você abrir um dataset ou usar o comando **describe** para ver o que um dataset contém, o label do dataset será mostrado.

- Variable labels: cada variável também tem um label de 80 caracteres, inicialmente em branco. Para utilizá-lo:
  label variable *varname* "*label*"
  Sempre que o STATA produzir algum output ele utilizará o label e não o nome real da variável.

- Value labels: definem um mapeamento entre dados registrados numericamente (variáveis devem ser numéricas) e as palavras que descrevem o que esses códigos numéricos significam. Inicialmente definimos o value label (ou seja, dizemos a STATA o que um determinado código numérico significa) e depois associamos esse value label a uma determinada variável que contém esses códigos. Note que é possível criar diversos value labels e é possível associar o mesmo value label a diversas variáveis.

  Os value labels são criados através do comando:
  label define *lblname* # "*string*" [# "*string*" ...] [, add modify]
  onde lblname é o nome do value label e # "string" corresponde ao código e seu label (ex.: label define msex 1 "masculino" 2 "feminino" 3 "ignorado").

  O comprimento máximo de lblname é 32 caracteres, e o comprimento máximo de string é 80 caracteres.

  A opção add permite adicionar um novo código a um value label já existene. Se add não é especificado, apenas novos value labels podem ser criados. Se add é especificado, você pode criar novos value labels ou adicionar novos códigos a um value label já existente.

  A opção modify permite a modificação ou deleção de um código já existente em um value label. Para eliminar algum código de um value label, é só ligar o código com uma string nula, "".

  Para associar um value label a uma variável usamos o comando (lembre-se que um mesmo value label pode ser atrubuído a diversas variáveis):
  label values *varname lblname* [, nofix]

  A opção nofix evita que o formato da variável seja aumentado para se igualar (se "fixar") ao comprimento do value label.

- Outros comandos para lidarmos com labels são:
  label dir = lista os mapeamentos existentes (as lblname)

  label list = lista o conteúdo de cada mapeamento (de cada lblname)

  label drop {*lblname* [*lblname ...*] | _all] = permite eliminar um, vários ou todos label values. Apesar de apagar o value label, ele continua associado a alguma variável. STATA não liga se uma variável está associada a um value label inexistente: ele age como se a variável não estivesse associada a nenhum value label. Para apagar a associação existente entre um value label (existente ou não) com uma variável, é só usar o comando label values sem definir um value label.

  label save [*lblname* [*lblname...*]] using *filename* [, replace] = permite gravar todos os comandos que definem um value label (label define) em um do-file.

labedit = esse comando refere-se a um ado-file preparado por John R. Gleason, Syracuse University, Syracuse NY, USA, cuja última versão foi publicada no STB-51 (insert dm56.1), que facilita a manipulação de labels. Se você tem essa ado-file, é só digitar labedit. Se não tem, veja item 1.9 e aprenda como baixar esse programa da internet).

1.15.8) Notes:

- O STATA também fornece a possibilidade de você escrever lembretes no dataset: gerais do dataset ou para cada variável. Esses lembretes funcionam como uma espécie de "post-it" eletrônico imbutido no dataset. A sintaxe é:

  notes [*varname*]: [TS] *text*
  notes
  notes [list] *evarlist* [in #[/#]]
  notes drop *evarlist* [in #[/#]]

  Na sintaxe, *evarlist* é uma varlist que pode conter a palavra _dta (significa que a nota é atribuída ao dataset em geral). TS, grava a data e a hora da nota. Até 9.999 notas podem ser atribuídas ao dataset e até 9.999 notas podem ser atribuídas a cada variável, sendo que o comprimento máximo de cada nota pode ser de 67.784 caracteres. Atenção: lembre-se se salvar o dataset, pois as notas são mantidas na memória e somente são realmente gravadas no dataset se ele for salvo. Sair do STATA sem gravar o dataset descarta todas as notas.

- O uso dos notes logs, comentários * em logs, é uma maneira muito eficiente de documentar toda a análise, desde a entrada de dados, até os resultados finais. Atenção: **USE SEMPRE** os recuros de logs, comentário e notes.

1.16) Funções, expressões, variáveis do sistema, acesso a coeficientes e resultados, explicit subscripting, time-series operators e problemas de precisão no STATA:

Referências: [U]16, [R]functions

1.16.1) Expressões e operadores:

- STATA permite uma grande gama de expressões, tanto aritiméticas quanto de string. Quando escrevemos expressões podemos diversos operadores.

  a) Operadores aritiméticos:

+	adição
-	subtração ou negação (valores negativos)
*	multiplicação
/	divisão
^	exponenciação

  Obs.: qualquer operação aritimética com um valor missing ou uma operação impossível (por exemplo, divisão por 0), irá retornar um valor missing. STATA avisa quantos missings foram gerados na operação.

  b) Operadores de strings:

+	concatenação

  c) Operadores relacionais:

>	maior que
<	menor que
>=	maior ou igual a
<=	menor ou igual a
==	igual a
~=	diferente de

  Obs.: expressões relacionais são "verdadeiras" ou "falsas" (na realidade são número 1 quanto true e 0 quando false: iso permite a realização de operações matemáticas com o uso de operadores). Os operadores relacionais podem ser usados tanto para números, strings ou missings. No caso de strings o que conta é a ordem alfabética, sendo que a capitalização também conta: as minúsculas são consideradas maiores (assim: zebra>casa, casa>Zebra, Casa<Zebra, casa<zebra, Casa<zebra, etc.)
  No caso de missings, um valor missing é maior que qualquer outro não missing (para observações importantes ver item 1.15.2).

  d) Operadores lógicos:

&	e
|	ou
~	não

Obs.: os operadores lógicos interpretam quanquer valor não zero (incluindo missing) como true (1), e zero como false (0) (também poder ser realizadas operações matemáticas com os operadores lógicos).

e) Ordem de avaliação dos operadores: a ordem de avaliação de todos os operadores é:
~, ^, - (negação), /, *, - (subtração), +, ~=, >, <, <=, >=, ==, &, |

Obs.: parênteses são permitidos e podem mudar a ordem de avaliação. Mesmo que não sejam necessários, é recomendado que sejam usados para facilitar a legibilidade.

1.16.2) Funções:

- Funções podem aparecer em qualquer expressão, e são indicadados por "função()". Todas as funções numéricas retornam missings quando trabalham com missings ou quando os resultados são indefinidos ou impossíveis. Aqui seguem apenas algumas funções mais usadas (para uma lista completa de todas as funções de todos os tipos, ver [U]16 e [R]functions).
- Funções matemáticas:

abs(*x*)	valor absoluto de um número (módulo)
exp(*x*)	exponencial
ln(*x*)	logarítmo natural
log(*x*)	logarítmo natural
log10(*x*)	logarítmo base 10
sqrt(*x*)	raiz quadrada
comb(n,k)	função cominatorial n!/{k!(n-k)!}

- Funções estatísticas:

Binomial(n,k,p)	binomial: probabilidade de observar k sucessos em n experimentos cada um com a probabilidade p de sucesso
chi2(df,x)	qui-quadrado: disbrituição cumulativa de qui-quadrado com df graus de liberdade
chi2tail(df,x)	qui-quadrado: cauda superior
F(df1,df2,F)	distribuição cumulativa F com df1 e df2 graus de liberdade no numerador e denominador
Ftail(df1,df2,F)	cauda superior da distribuição cumulativa F
normden(z)	densidade padrão N(0,1)
normden(z,s2)	densidade padrão N(0,s2) (s2 = variância)
norm(z)	normal cumulativa padrão
ttail(df,t)	distribuição t reversa: retorna probabilidade de T>t
uniform()	números pseudo-aleatórios uniformemente distribuídos

Obs.: uniform() retorna números pseudo-aleatórios uniformemente distribuídos no intervalo [0,1). Não toma argumentos, mas os parênteses devem ser digitados. Para maiores detalhes, consultar [U]16 e [R]generate (item set seed)

- Funções de strings:

substr(s,n1,n2)	retorna a substring de s começando no caractere n1 correndo por n2 caracteres. Se n1<0, n1 é interpretado como a distância a partir do final da string; se n2 =., corre por todo o final da string a partir de n1
string(n)	converte números para strings
string(n,s)	converte números para strings mantendo os formatos (%fmt)
real(s)	converte strings para números ou .
abbrev(s,n)	retorna s, abreviado para n caracteres
index(s1,s2)	retorna um número correspondente a posição em s1 no qual s2 foi achado pela 1ª vez
subinstr(s1,s2,s3,n)	retorna s1 no qual as primeiras n ocorrências de s2 foram substituídas por

	s3. Se n = ., todas as ocorrências são substituídas.
subinword(s1,s2,s3,n)	retorna s1 no qual as primeiras n ocorrências de s2 como uma palavra (isolada por espaços) foram substituídas por s3. Se n = ., todas as ocorrências são substituídas.
lower(s)	torna string para minúsculas
upper(s)	torna string para maiúsculas
ltrim(s)	retira espaços em branco do começo da string
rtrim(s)	retira espaços em branco do fim da string
trim(s)	retira espaços em branco do final e do início da string
match(s1,s2)	retorna 1 se s1 "matches" o padrão de s2, senão retorna 0

- Funções de datas: s é usado para significar uma subexpressão de string, e, m, d e y são usados para indicar uma subexpressão numérica.

date("s1","s2"[,##y])	retorna a elapsed date correspondendo a s1 baseada em s2 e y. s1 contém a data, registrada como string, em virtualmente qualquer formato (meses podem ser por extenso, abreviados ou indicados por números; anos podem incluir ou excluir o século; pode ter brancos ou qualquer pontuação). s2 é qualquer permutação de m, d, e y, sendo que sua ordem define a ordem no qual o dia, mês e ano estão registrados em s1. ##, se especificado, indica o século padrão para anos com 2 dígitos em s1. y fornece uma alternativa para lidar com anos de 2 dígitos, pois y especifica o maior ano que deve ser retornado quando um ano de 2 dígitos é encontrado. Se ## nem y forem especificados, date() retorna um valor missing quando encontra um ano de 2 dígitos. Sintaxe completa da função: date(["]s1["],"[##]s2",[,##y]) (ver [U]27).
mdy(m,d,y)	retorna a elapsed date correspondendo a m, d e y.
day(e)	retorna o dia numérico do mês correspondente a e
month(e)	retorna o número do mês correspondente a e
year(e)	retorna o número do ano correspondente a e
dow(e)	retorna o dia da semana correspondente a e
doy(e)	retorna o dia numérico do ano correspondente a e
week(e)	retorna o número da semana do ano
quarter(e)	retorna o número do trimestre do ano
halfyeary(e)	retorna o número do semestre

- Funções de time-series: STATA tem diversas funções para lidar com time-series. Para maiores detalhes ver [U]16.3.4 e [U]16.8

Funções especiais: São funções especiais do STATA para lidar com aspectos não cobertos com as outras funções:

int(x)	parte inteira de um número
sum(x)	soma os valores de x, tratando missing = 0
r(name)	valor do resultado salvo de nome r(name). Útil para acessar os resultados calcudados por <u>comandos gerais</u> do STATA
e(name)	valor do resultado salvo de nome e(name). Útil para acessar os resultados calculados por <u>comandos de estimação</u>
s(name)	valor do resultado salvo de nome s(name). Útil para progra madores para interpretar comandos que são digitados
float(x)	útil se suas variáveis numéricas são do tipo

float, pois corrige o erro que o STATA retorna quando compara dados float em expressões numéricas tipo if==

inrange(z,a,b) retorna 1 se a<=z<=b (vale para números ou strings)

missing(exp) retorna 1 se exp avalia para missing

1.16.3) Variáveis do sistema:

- As expressões também podem conter _variáveis (variáveis subscritas), que são variáveis internas criadas e mantidas pelo STATA. Todas as variáveis subscritas têm o nome iniciando por um subscrito. As _variáveis são:

  [*eqno*]_b[*varname*] contém o valor do coeficiente de varname obtido no modelo mais recentemente estimado pelo STATA

  [*eqno*]_coef[*varname*] sinônimo do anterior

  [*eqno*]_se[*varname*] contém o valor do erro padrão de varname obtido no modelo mais recentemente estimado pelo STATA

  _cons é uma constante sempre igual ao valor 1 quando usada diretamente ou igual ao intercepto quando usada indiretamente, como por exemplo em _b[_cons] ou _se[_cons]

  _n contém o número da observação atual

  _N contém o número total de observação no dataset

  _pi contém o valor de pi

  _rc contém o valor do return code mais recente

1.16.4) Acessando coeficientes, erros padrão e resultados de comandos:

- Após estimar um modelo, você pode acessar os coeficientes e os erros padrão e usá-los em expressões subseqüentes. O modo de acesso é diferente dependendo se: é um modelo simples, se é um modelo de ANOVA ou se é um modelo de múltiplas equações.
- Modelos simples: Um modelo simples é aquele que fornece uma única equação com uma correspondência um-para-um entre os coeficientes e cada variável. Para acessar esses coeficientes, devemos usar as variáveis de sistema (ver 1.16.3):

  _b[*varname*] acessa o coeficiente de cada variável

  _se[*varname*] acessa o erro padrão da variável

  _b[_cons] acessa o coeficiente da constante

  _se[_cons] acessa o erro padrão da constante
- Modelos de ANOVA: nesse tipo de modelo não há uma relação simples entre os coeficientes e as variáveis. Para variáveis contínuas, podemos usar _b[*varname*], mas para variáveis categóricas devemos especificar o <u>nível</u> e a <u>variável</u>. Por exemplo: _b[drug[2]] refere-se ao coeficiente do segundo nível da variável drug. Para interações, _b[drug[2]*disease[1]] refere-se ao coeficiente do segundo nível da variável drug e o primeiro nível da variável doença. Os erros padrão são obtidas da mesma forma, apenas usando-se a variável de sistema _se[].

- Modelos com múltiplas equações: a sintaxe é a seguinte:

  [*eqno*]_b[*varname*] ou [*eqno*]_coef[*varname*]

  [*eqno*]_se[*varname*]

  Nesse caso [eqno] refere-se ao número da equação estimada, por exemplo: [#3]_se[disp] refere-se ao erro padrão da variável disp na 3ª equação estimada e [#1]_b[disp] refere-se ao coeficiente da variável disp na 1ª equação estimada.

1.16.5) Acessando os resultados de comandos do STATA:

- Todos os comandos no STATA (não apenas os de estimação) salvam temporariamente seus resultados de uma maneira que podemos acessar esses resultados e utilizá-los em qualquer expressão que quisermos.
- Os comandos do STATA são classificados em 4 tipos:

  r-class salvam os resultados dos comandos gerais

  e-class salvam os resultados dos comandos de estimação

  s-class salvam os resultados de comandos de passagem (para programadores)

  n-class não salvam comandos

- Para visualizar quais resultados são salvos em cada comando utilizamos a sintaxe:
  return list mostra os comandos r-class salvos
  estimates list mostra os comandos e-class salvos
  sreturn list mostra os comandos s-class salvos
- Para utilizar os resultados salvos, usamos 3 funções especiais (ver 1.16.2):
  r(*name*) utiliza os comandos salvos em r()
  e(*name*) utiliza os comandos salvos em e()
  s(*name*) utiliza os comandos salvos em s()
- Por exemplo: podemos usar o comando summarize age. Para visualizar quais os r() que foram salvos utilizamos return list. Para utilizar os comandos é só escrever a expressão, por exemplo: generate zage = (age - r(mean))/r(sd) (geramos uma variável padronizada para a variável age).
- Atenção: se você pretende utilizar algum comando salvo em r(), e() ou s(), deve utilizar logo após o comando que o gerou, pois o próximo comando no STATA apagará e/ou salvará outros resultados. Lembre-se que são resultados salvos temporariamente.

1.16.6) Explicit subscripting:

- Observações individuais em cada variável pode ser referenciadas "subscritando-se" as variáveis. Explicit subscripting são especificados pelo nome da variável e colchetes que contém as informações e expressões de subscripting. O resultado da expressão de subscripting é truncado para um inteiro, e o valor da variável para a observação indicada é retornado. Se o valor da expressão de subscripting é menor do que 1 ou maior do que _N, um valor missing é retornado.
- O explicit suscripting funciona da seguinte maneira: quando você digita algo como generate y = x, STATA entende generate y = x[_n], ou seja: o primeiro valor de y é igual ao primeiro valor de x, o segundo valor de y é igual ao segundo valor de x, e assim por diante. O que garante essa seqüência é o subscrito (nesse caso subentendido) _n (que numera as observações - ver 1.16.3). Se você explicitar o subscripting, como por exemplo em: generate y = x[1], todas as observações de y serão iguais ao valor da 1ª observação de x; generate y = x[45] torna todas as observações de y iguais ao valor da 45ª observação de x; generate y = x[_N] torna todas as observação de y iguais ao valor da última observação de x.
- Quando um comando é precido de by...:, prefixos, expressões suscritas e as variáveis subscritas _n e _N são avaliadas em relação ao subconjunto de dados correntemente sendo processado pelo by, ou seja, o by-group atual.
- Atenção: **APRENDA A USAR** o uso de explicit subscripting, das variáveis de sistema _n e _N, e do by...:, pois geralmente são a solução final para expressões complexas e difíceis. Para maiores informações ver: [U]16.4, [U]16.5, [U]16.6 e [U]16.7.

1.17) Do-files:

Referências: [GSW]15, [U]19

- Um do-file é um arquivo em formato ASCII, com a extensão .do, que contém uma seqüência de comandos para o STATA executar. Todos os comandos do STATA podem ser colocados em um do-file. É semelhanto ao .pgm do Epi Info.
- Para criar um do-file pode ser utilizado qualquer processador de texto, desde que o formato gravado seja ASCII e você tenha o cuidado de que a extensão seja .do (pode ser necessário usar aspas para fixar a extensão). É mais fácil usar o STATA Do-File Editor, que já escreve em ASCII e grava com a extensão .do.
- O Do-File Editor tem diversas ferramentas úteis na elaboração das do-files nas suas barras de menu e ferramentas, incluindo: inserir do-file, search, balance/match parênteses, mudança de capitalização, etc.
- Ao criar do-files, é importante colocar na 1ª linha a versão do STATA no/para o qual o do-file foi criado. Isso é feito através do comando **version 7.0,** por exemplo. Isso garante que o do-file continue funcionando mesmo em versões superiores do STATA.
- Comentários nos do-files são escritos começando a linha com um *, da mesma forma que os comentários nos logs. Também podemos incluir comentários no meio

dos comandos através do uso de ***/* comentário */***

- Linhas longas podem ser "quebradas" com o uso da sentença: **/**/**, para melhorar a legibilidade. Por padrão, o delimitador de final de linha é o "enter", ou seja, o "carriage return" (cr). Também podemos mudar o delimitador do final de linha para ponto e vírgula (;) através do comando **#delimit {cr|;}**. Se mudarmos o delimitador de final de linha para ; temos que nos assegurar de não esquecer de colocar um ; no final de cada linha de comando. Se o delimitador é o cr, para quebrar as linhas é melhor usar o /**/.
- Um do-file completa sua execução quando: a) chega ao final dos comandos; b) encontra um exit; ou c) um erro ocorre. Se ocorrer um erro, o final dos comandos não é executado e o STATA apresenta 2 mensagens de erro: a primeira refere-se a mensagem de erro que ocorreu quando executava determinado comando, e a segunda informa que o STATA terminou o do-file pelo mesmo motivo.
- Um do-file pode chamar outro do-file, que pode chamar outro do-file, e assim por diante. STATA permite chamadas desse tipo com até 32 do-file de profundidade. Note bem: você poder ter infinitas chamadas a do-files, mas cada uma só pode ter 32 do-files de profundidade.
- Para rodar um do-file:
  do *do-file_name* [, nostop] executa o do-file normalmente
  run *do-file_name* [, nostop] executa o do-file mas <u>não mostra</u> nenhum output
  A opção [, nostop] diz que o do-file deve ser executado até o fim, mesmo se tiver erros.
  Também podemos rodar um do-file em batch mode (ver 1.3.3): devemos ir ao diretório de trabalho e executar o comando:
  c:\stata\data> **c:\stata\wstata.exe [/m#] /{s|b} do *do-file***
- As do-files também podem ser usadas para programação. Para maiores informações ver texto em [U]19.6.

1.18) Ado-files:

Referências: [U]20

- Uma ado-file define um comando do STATA, mas nem todos os comandos do STATA são ado-files. Quando por exemplo você usa o comando summarize, você está usando um comando interno do STATA. Quando você usa o comando ci, você está usando um comando programado em uma ado-file. Os resultados de ambos são indistinguiveis.
- Uma ado-file é um texto em ASCII que contém um programa para o STATA. Quando você digita o nome de um comando não interno do STATA, ele procura em determinados locais por uma ado-file com esse nome: se ele encontra, executa o programa determinado pela ado-file.
- Qualquer pessoa pode programar um comando, uma análise, qualquer coisa, no STATA e deixar o programa como uma ado-file. Também podemos escrever uma arquivo de ajuda com a sintaxe, opções e exemplos de sua ado-file, que será acessado por qualquer comando de busca, como por exemplo search. Por exemplo: o commando para desenhar diagramas de Venn não foi fabricado pela STATA Coorporation: foi feito por JM Lauritsen e chama-se venndiag.ado.

- Para saber se um comando é interno ou uma ado-file, use o comando **which *command_name*.**
- A programação de qualquer ado-file pode ser visualizada utilizando um processador de texto comum ou o Do-File Editor.
- Quando digitamos um comando que é um ado-file, STATA procura em seus diretórios padrão, que podem ser conhecidos através do comando **sysdir**. Geralmente são:

STATA	c:\stata\
UPDATES	c:\stata\ado\updates\
BASE	c:\stata\ado\base\
SITE	c:\stata\ado\site\
STBPLUS	c:\ado\stbplus\
PERSONAL	c:\ado\personal\
OLDPLACE	c:\ado\

A função desses diretórios é a seguinte:

STATA	Diretório principal do sistema
UPDATES	Armazena as atualizações oficiais das ado-files (fornecidas por STATA)
BASE	Armazena as ado-files oficiais básicas que vêm no CD do STATA
SITE	Armazena as ado-files que o administrador de uma rede quer distribuir

PERSONAL Armazena as ado-files que você mesmo escreveu
STBPLUS Armazena as ado-files distribuídas pelo Stata Technical Bulletim ou outras fontes (como outros web sites, disquetes, etc.)
OLDPLACE É para usuários de STATA antigos. Antes do STATA 6.0, todas as ado-files pessoais são escritas no mesmo diretório - OLDPLACE.

No geral esses diretórios podem ser categorizados da seguinte maneira:

Diretórios oficiais: UPDATES e BASE: ado-files oficiais originais ou atualizações
Diretórios pessoais: PERSONAL, STBPLUS, OLDPLACE: ado-files que você instala ou programa
Diretórios públicos: SITE: para ado-files públicas em rede de computadores

1.19) Comandos imediatos e display:

Referências: [U]22

- Um comando imediato é um comando que não obtém os dados do dataset na memória, mas sim, dos números que você digita no teclado. Esses comandos tornam o STATA uma glorificada calculadora de mão (para cálculos comuns, simples, complexos, epidemiológicos, etc.).
- Os comandos imediatos nunca interferem nos dados da memória e sua sintaxe é semelhante em todos: o nome do comando seguido por números, que são as estatísticas sumárias a partir da qual o comando é executado.
- Todos os comandos imediatos terminam com a letra i, embora o inverso não seja verdade.
- Uma breve lista dos comandos imediatos:

bitesti	Teste de probabilidades binomiais
cci	tabelas para epidemiologistas
csi	idem
iri	idem
mcci	idem
cii	intervalo de confiança para médias, proporções e contagens
prtesti	testes de proporções de uma ou duas amostras
sampsi	determinação de tamanho de amostra e poder estatístico
sdtesti	testes de comparação de variâncias
symmi	testes de simetria e homogeneidade marginais
tabi	tabelas de freqüência de uma ou duas vias
ttesti	teste de comparação de médidas

- Existe ainda um comando especialmente feito para ser usado como uma calculadora: o comando **display exp**, onde exp é qualquer expressão permitida por STATA.